Sample Configuration for swtorssh.cfg

The next 6 lines of text, shows us,how a valid configuration file swtorssh.cfg should looks like.

```
fullssh.sh  ssh-id Compress 4 2 22  9999                    noshell username1@dnsname.server.de    xxxxx xxxxxx Germany     my-home-server-wan
fullssh.sh  ssh-id Compress 4 2 22  9999                    clock   username1@10.0.0.66            xxxxx xxxxxx Germany     my-home-server-lan
fullssh.sh  ssh-id xxxxxxxx 4 2 22  9999:172.29.255.1:8080  noshell username2@127.124.166.21       xxxxx backup Austria     freds-server
fullssh.sh  passwd Compress 4 1 443 9999                    noshell hjfdj@84.55.22.166             xxxxx xxxxxx France      french-pgp-server
chainssh.sh ssh-id Compress 4 2 22  9999                    chain   username1@dnsname.server.de    11000 xxxxxx Canada      mychain1.cfg
chainssh.sh ssh-id Compress 4 2 22  9999                    chain   ujdfhj1@188.67.88.15           11000 xxxxxx Poland      mychain2.cfg
```

This help file describes the syntax for the configuration file

~/Persistent/swtor-addon-to-tails/swtorcfg/swtorssh.cfg

This settings are all valid with version 0.83 or higher.

- Every possible ssh-connection is defined in one single line.
  In this example above are 6 remote ssh-servers defined.
- There is no limitation of ssh-servers, that can be defined.
- Every ssh server could be defined with a valid ipv4 address or a user friendly dns-name.
- All configuration parameters inside the configuration file swtorssh.cfg are
  case sensitive.
- There are 13 columns possible for every ssh-connection. All 13 fields have to be filled out !
- The same combination of username and host can be used more than once, if the port is different !
- Spaces “ “ inside the of the fields 12 and 13 are not supported ! Every space “ “ has to be replaced with a “-”.

Column 1:

This column 1 has 3 possible entry's → [fullssh.sh] [chainssh.sh] [pfssh.sh]
Only one of them, can be chosen in a definition of a remote ssh-server.

fullssh.sh

This the best possible option for a ssh-connection. In this case, we have somewhere a ssh account without any restrictions or borders for our own actions.

- A login over password or a login over predefined private and public keys is possible.
- We have enough space to store some backups on that ssh-server.
- We have a unrestricted shell to use like /bin/bash.
- We could use the ssh commands scp or sshfs to transfer files to the remote ssh-host.
- We can use rsync over SSH to make a Backup of our Persistent Volume.
- We could even start a remote X11 applications over the ssh-tunnel like xclock or even a browser if we have enough speed.

Warning : There are some “fullshell” Servers on the Net that prevents the starting of X11 applications over SSH. In this particular case you can’t start a little X11 Clock to show the current time from the remote SSH-Server. This would mean option “noshell” for this connection.

If you are searching for very fine SSH Host that will have all of the above options possible for the remote users, even if they use Tails to connecting ... have a closer look on this URL.

https://www.freeshell.de

Note : Since September 2024 this host will frequently used by hackers and therefore the ssh-service is sometimes down !

chainssh.sh

A few SSH-servers on the internet don't like direct any connection attempts made from over the onion-network.
If we would like to connecting to such a onion unfriendly ssh-server, we have to first masquerade our origin IP with a first ssh server in place,acting as a jump-server. You have to connect to the jump Server first and from over there without any password to the second ssh-server itself.

tails -> onion-network Nodes 1,2,3 -> ssh-connection > JUMP-SERVER -> ssh-connection -> DESTINATION-SERVER

This scenario is bit a tricky to accomplish,because of the following reasons.

- The desired DESTINATION-SERVER has to support a pass-wordless login with a public key.
- A login with password on the final SSH-Server is not supported.
- There are at least 2 different public/private ssh keys involved into this configuration.
- Column 11 on the same line contains the configuration connect the destination from our first jump-server.
- This communication is very slow. We are adding 5 additional so called “hops” until we can contact the Website we would like to visit.

pfssh.sh

This kind of shell-account is very limited in the usage in general. As a little example, who offer this kind of a limited ssh-access accounts, you may have a look on this following website. This kind of account may is the right choice, if you need to create a valid SSH-connection within minutes.

https://www.dewassh.net/server/SSH
https://www.xshellz.com/
https://www.vpnjantit.com/

This very limited accounts can be fast and easily created within minutes, with a few very drastic downsides in general for using it.

- Only a simple port-forwarding is allowed,nothing more than this simple task can be done.
- Multiple logins with the same username are not supported.
- No other shell than the good old “jail” called /bin/false is usable over this very limited ssh connection. This means you can only create a remote socks5 proxy,nothing more. You can’t even execute any command on the remote server over SSH. Right here you see a example how it looks inside swtorssh.cfg

```
pfssh.sh passwd Compress 4 2 143 9999  noshell LIiS-dewassh.net@us.serverkit.me   xxxxx           xxxxxx  USA          3-Day-account
```

- A login is only possible with a valid password. Logins over a public/private key are not possible or even not desired.
- The created SSH-logins has a expiration time between 3 and 30 days. After the expiration of the created account, you have to create a new one.

- Some really silly and insane ssh-providers (others than dewash.net) only allow the creation of a ssh-account with a valid Facebook or a Google Email account. Of course, we don’t give them away our credentials for Facebook or even a fake Google Email Account. We use a fake Email, if we have to register for a account with a Email.

- You can not contact the Website www.dewassh-net very easy with the standard TOR-Browser from Tails. If you really have no other choice as with the TOR-Browser, you need at least 20 min. time to go through this “captcha-terror”, because the remote system detect us as a user of Tails or TOR-Browser.

- I would highly not recommend you to create this login over your regular Windows or Linux Computer. You know the reason. The public WAN Address of the ISP could be tracked back to you as a person ! It may sounds better to use a puplic VPN to register only for the creation for a SSH-account with dewash.net. If you already have working SSH-connection over the add-on you can use this connection.

Column 2:

This column 2 has 2 possible entry's → [ssh-id] [passwd]
Only one of them, can be chosen by a single line.

ssh-id

If we have a private/public key generated for a ssh-connection,we can use this public key stored inside ~/.ssh as a replacement for a password-less login to the remote server.

passwd

Every time we connect to this ssh-server, we have to provide a valid password. We have to type the password over the keyboard every time.
The password will be send over to the remote ssh-server with sshpass. The password for this session is not stored on the persistent volume. The password will be asked in a dialog-box on every startup of the ssh-connection.

Column 3:

This column 3 has 2 possible entry's → [Compress] [xxxxxxxx]
Only one of them, can be chosen.

Compress

By default compression is not enabled for any outgoing ssh connections on any Linux system. Very simple compression is no problem for
a modern CPU and even on a gigabit network it is faster to compress data first before sending it over the network instead of leaving it uncompressed.
This options tells the scripts that we would like to use Compression in our specific ssh-connection. On the ssh command-line of Linux this would be
the option -C.

xxxxxxxx

In the case that no Compression could be used, this dummy text has to be used.

Column 4:

This column 4 has 1 possible entry → [4]

Tails only supports TCP/IP Version 4. Tails doesn’t support IP Version 6 or any kind of outgoing UDP IP V4 connection.

Column 5:

This column 5 has 2 possible entry's → [2] [1]
Only one of them, can be chosen.

There are today only 2 ssh protocols available to use inside of ssh. Most users should write a 2 here. Only use the older and not longer used ssh
protocol version 1,if you know exactly, what you are doing.

http://www.snailbook.com/faq/ssh-1-vs-2.auto.html

Column 6:

This column 6 has only one possible entry → [22]

It defines the remote TCP port of the SSH-server we would like to connecting. The standard port for a standard ssh server
is 22. This port information may can be something different than 22.

Column 7:

This column 7 has only one possible entry → [9999]

This value describes the local port to build our socks5 server. The only possible value is 9999. You could change this value
if you would like to change it, but this predefined port 9999 is hard encoded into all the scripts. Do change this value
to something other than 9999 only, if you know what you are exactly doing !

If you would like to use a already established socks5 server on the remote ssh server, you can write the following line

99999:172.21.255.1:8080

Warning : Only with the new Release 0.83 it is possible to use this feature called “ssh port redirection”. In older releases the only
valid entry is 9999. The socks5 server has to be exist on the remote server. Only a internal IP of the remote Server can be used here.

```
Chromium with socks5 on 127.0.0.1:9999
            |
[Tor System Local Port 9999] -------> [t0r1]---> [tor02] --→ [tor03] --------→ [Remote SSH 1]
                                                                                 172.21.255.1 ---------------> [Remote SSH2] ---→ Internet
```

This configuration is almost identical to a jump-server, but in this configuration you can use any form of connecting to the remote
Remote SSH2 Server.

Column 8:

This column 8 has 3 possible entry's → [noshell] [clock] [chain]
Only one of them, can be chosen.

noshell

After a successful login we could normally start a remote shell on the foreign ssh-server. Because we are in no need to use
a classic shell after the login this option is used. On a Linux server that shell would be normally /bin/bash.
Please remember to write exactly noshell and not NOSHELL or Noshell. This is the only entry that works with pfssh-mode.

clock

Almost every ssh server can make use of the X11-window system. If the statement clock is found here, a nice clock like the
following one as example

http://tclock.sourceforge.net/tclock-shot.gif

is displayed as soon we are connected to our SSH server. This option clock needs a fullssh.sh connection inside of the field 01.
The displayed clock on our Tails-Screen shows the current time of the remote ssh-server. Some servers do provide us with a full shell
system but prevent us to starting X11 applications like the clock described. In this case only “noshell” is the only option for this field.
If you are the admin or owner of your own SSH Server please provide the following line in your SSH-Server configuration.

X11Forwarding yes

chain

With this option it is possible to build a "JUMP-SERVER" configuration. This special keyword is mandatory, because we need to connect to
a second ssh-server after successful login on the first server.

Column 9:

This column 9 has only one possible entry → username@server

The data provided here, containing the username and the host-server we would like to connecting over ssh.
A possible entry of this field with a DNS server-name:

username1@dnsname.server.com

or a the same entry with a IP instead of a user friendly dns-name :

username1@177.56.43.3

Column 10:

This column 10 has 2 possible entry's → [xxxxx] [11000]
Only one of them, can be chosen. This field is only used by a “chainssh.sh” configuration.

xxxxxx

Almost all users can write this dummy text xxxxx here.

11000

This single value here written in this field can be in the range of
1024 until 65535.A written value here is only valid in the main mode "chainssh.sh" of field 01,
we need this value as second port definition.

Column 11:

This column 11 has 2 possible entry's → [xxxxx] [backup]
Only one of them, can be chosen.

xxxxx

Almost all users can write this dummy text xxxxx here.

backup

If you have multiple SSH-servers to chose from, with this column 11,you can set exactly **one** of this ssh-server from the list
as a predefined backup-server. Multiple backup statements inside of the configuration file swtorssh.cfg are not supported by now.
After Release 0.60 it is possible to encrypt the complete backup. You can only transfer the backup to a remote host if you decide
to encrypt it during the backup. With this add-on it is not possible to transfer a simple tar.gz backup to a other Host.

AS A VERY IMPORTANT REMINDER AGAIN :
DO NOT COPY A NOT ENCRYPTED BACKUP (a tar.gz file) FILE FROM YOUR PERSISTENT VOLUME TO A SSH-SERVER YOU DON’T TRUST FOR ABOUT 100 %.
BECAUSE THIS BACKUP CONTAINS VERY SENSITIVE DATA (your personal ssh-keys and your Bitcoin Wallet only as a Example) !!!

Column 12:

This column 12 describe the country, that our ssh-server is located. Please write a single "-" instead of a single space " " if your complete
country-name contains spaces inside of the name.

United Kingdom will be to ->United-Kingdom
Dominican Republic will be -> Dominican-Republic

Of course if we are using the mode “chainssh.sh” we do provide the country where our server2 is located.

```
----------
Column 13:
----------

This column 13 is a description field for most users with one exception, if you are using chainmode.ssh.
Like in the country of field 12 write a single "-" instead of single space " " inside of the text as description.

my-home-server-wan
my-home-server-lan

In the main mode "chainssh.sh" only, we use this description field to make our second connection. We need to create a new .cfg file for every
chainmode.sh connection, we would like to use inside of the add-on. If Columns 13 contains the value mychain1.cfg.
You should create a file called mychain1.cfg inside of the directory ~/Persistent/swtor-addon-to-tails/swtorcfg
In this example we could write the following single command inside this configuration file mychain1.cfg

ssh -o ServerAliveInterval=10 -AtC42N -D11000 fs-username1@war.freeshells.org

This command written here, will be executed on our “JUMP-SERVER” after we connected to make the ssh-connection to the Destination server
in our chain.


- ServerAliveInterval=10     Keep the ssh connection from host1 to host host2 open

- A                          Enables forwarding of the authentication agent connection
- t                          Force pseudo-terminal allocation
- C                          Requests compression of all data
- 4                          only use tcp version 4
- 2                          Use only ssh 2 protocol
- N                          Do not execute a remote command.  This is useful for just forwarding ports.
- D                          Specifies a local “dynamic” application-level port forwarding

Note : -D11000 must be made, according the port definition inside the value of column 10.
Numbers below of 1023 can not be used,because only the user root can use this low sport numbers.


Please note the following :

In chain-mode we need at least 5 hopes [ node1 – node2 – node3 – jumpserver – destination-server] until the
packets go to the journey …

This communication is may very slow .. but also very secure to hide you identity.


Note : Since release 0.83 you may better use “port redirect” feature of the script. If you are using port-redirect on your own server you
have a few better control than in chainmode.sh.
```